

Bien débiter en C++

par Alp Mestan ([Site personnel](#)) ([Blog](#))

Date de publication :

Dernière mise à jour :

Vous souhaitez apprendre le C++ mais ne savez pas comment vous y prendre, par quoi commencer ? Cet article est fait pour vous ! Des outils les plus adaptés jusqu'aux meilleures ressources pour apprendre, cet article vous donnera toutes les pistes nécessaire pour effectuer un apprentissage correct du C++.

Commentez

I - Qu'est-ce que le C++ ?.....	3
II - Les livres C++.....	3
III - Les cours en ligne sur le C++.....	3
IV - Ce n'est pas suffisant ?.....	3
V - Les outils C++.....	4
VI - Il me reste des questions.....	4
VII - Les bibliothèques.....	4
VIII - Je voudrais aller plus loin !.....	5
IX - Conclusion.....	5

I - Qu'est-ce que le C++ ?

Le C++ est un langage de programmation très répandu et très utilisé : on s'en sert pour réaliser des programmes embarqués dans des appareils mobiles comme pour des jeux vidéos, des logiciels de gestion de données comme des compilateurs pour d'autres langages de programmation. Le C++ est un langage où l'on peut plus ou moins tout faire, et de différentes façons, ce qui fait sa puissance mais en le rendant de fait plus difficile à apprendre.

C'est au début des années 80 que Bjarne Stroustrup a inventé le "C with classes" : à ce moment-là, il ne s'agissait que d'un C auquel on avait rajouté des classes et de l'héritage. Depuis plus de 20 ans, le langage C++ a énormément évolué et il a été normalisé. Aujourd'hui, C et C++ sont deux langages très, très différents bien qu'ils partagent un peu d'histoire. C'est pourquoi dans cette page nous vous proposons des ressources qui vous permettront d'apprendre le C++ "moderne", et non plus en tant que langage qui hérite du C avec un apprentissage du C au préalable. En effet, il est même déconseillé pour un débutant d'apprendre le C avant le C++ si c'est pour apprendre le C++ au final.

Le C++ est un langage multi-paradigmes : objet, fonctionnel, générique sont très communs en C++. Il est d'une richesse incroyable mais il faut connaître au mieux les règles du jeu pour écrire des logiciels solides, que l'on pourra faire évoluer facilement.

II - Les livres C++

Il n'existe pas **un** livre C++ qui convient à tous les profils, en revanche, nous vous en présentons ici selon différents critères, afin de choisir celui qui vous conviendra le mieux et qui rendra votre apprentissage le plus efficace possible. Vous trouverez une liste de livres C++ sur [la page livres C++](#).

- Si vous n'avez jamais programmé avant, ou presque pas, nous vous conseillons **C++ je me lance** de Francis Glassborow, bien qu'il date un peu il s'avère très efficace pour les débutants en programmation. Il se complètera ensuite très bien avec un ouvrage de référence sur le C++.
- Si vous avez déjà programmé et que vous lisez l'anglais, vous pouvez vous tourner vers **Accelerated C++** de Koenig & Moo pour une introduction rapide ou **C++ Primer, 4ème édition** pour une introduction plus souple.

III - Les cours en ligne sur le C++

Il y a un certain nombre de **cours en ligne** de bonne qualité. Certains utilisent l'approche historique : on apprend le C++ en tant que sorte "d'évolution" du C, qui n'est plus l'approche appréciée aujourd'hui. De plus, un cours, aussi bon soit-il, ne remplacera pas un bon livre.

Deux des cours hébergés sur Developpez sont assez complets, malgré le fait qu'ils utilisent l'approche historique (ce qui vous demandera de garder un certain recul par rapport aux pratiques du C qu'ils peuvent reprendre) : le **méga-cours** de Christian Casteyde ainsi que **Penser en C++** (qui est la traduction d'un livre) de Bruce Eckel. Je recommanderais personnellement *Penser en C++* que je trouve légèrement plus correct et précis, mais ces deux cours vous formeront très convenablement aux bases du C++, vous permettant de vous reposer sur de bonnes bases pour la suite de votre apprentissage.

Pour terminer, quelles que soient les sources où vous puiserez vos informations, il vous faudra garder un certain recul et garder le meilleur de chacune car *La vérité n'est nulle part*.

IV - Ce n'est pas suffisant ?

Non, un cours et/ou un livre sur le C++ ne sont pas suffisants. En effet, ces derniers répondent au problème du "comment" (comment écrire un logiciel qui fait ceci ou cela), alors qu'il y a aussi le problème du "pourquoi". Il vous faudra impérativement, si vous êtes un débutant total en programmation, vous munir d'un ou plusieurs ouvrages et/ou cours sur :

- L'algorithmique et les structures de données : il vous faudra souvent, lorsque vous programmez, stocker des données de manière intelligente et efficace, mais surtout appropriée à l'usage qui en est fait. Connaître les structures de données les plus utilisées ainsi que leurs caractéristiques est un élément très important. De même, connaître les bases de l'algorithmique (variables, boucles, complexité, invariants, ...) est tout aussi nécessaire pour que vous écriviez de bons programmes. Vous pouvez vous référer à la page [Algorithmique](#) pour ce faire ;
- La conception logicielle : il vous faudra savoir organiser intelligemment les différentes parties de vos programmes de sorte à pouvoir facilement introduire des évolutions dans ce dernier, par exemple. Pour ce faire, il y a également une certaine connaissance à acquérir et à mettre en pratique. Vous pouvez vous référer à la page [Conception](#) pour ce faire.

V - Les outils C++

Le C++ existe depuis plus de 20 ans. Par conséquent, il y a une quantité phénoménale d'outils pour écrire du code C++, le compiler, le déboguer, le documenter, l'analyser, etc. Voici donc les recommandations de l'équipe à ce sujet. Ne sont présentés ici que les outils gratuits ; vous pouvez trouver une liste plus complète et incluant les produits payants sur la [page outils](#).

- Environnement de développement intégré (compilateur + éditeur de code source, au minimum, qui vous mâche le travail dont vous n'avez en général pas à vous soucier) : [Microsoft Visual C++ Express](#) (Windows seulement) ou [Code::Blocks](#) (Windows, Linux, Mac) ;
- Editeur de texte (permet simplement d'écrire du code, souvent avec de la coloration syntaxique) : [Notepad++](#) (Windows), Editeurs de texte pré-fournis comme Emacs, VI(M), Kate, Gedit (Linux - selon la distribution) ;
- Compilateur (transforme du code C++ en code compréhensible par votre ordinateur) : [Visual C++](#) (Windows), [MinGW](#) (Windows), [Intel C++ Compiler](#) (Windows, Linux) ou [g++](#) (Linux) ;
- Débogueur (permet d'inspecter en détails l'exécution de vos programmes, trop peu utilisé par les débutants bien que très utile) : celui intégré à Visual C++ (Windows), [GNU GDB](#) (Linux) ;
- Générateur de documentation (génère un ensemble de fichiers HTML depuis votre code ainsi que les commentaires qui l'accompagnent) : [Doxygen](#).

Il vous est conseillé, dans un premier temps, d'utiliser un éditeur de code et le compilateur à côté. En effet, cela vous évitera de nombreux problèmes une fois que vous entamerez des manipulations plus compliquées de connaître, au moins dans les grandes lignes, le fonctionnement de la compilation d'un programme C++, les options essentielles du compilateur, etc.

VI - Il me reste des questions...

Alors vous avez bien appris grâce à un bon livre et/ou un bon cours, mais il vous reste des questions ? Dans ce cas, nous vous recommandons d'aller consulter la [Foire Aux Questions C++](#) qui aborde des sujets divers et variés à propos du C++, depuis les petites astuces de base jusqu'à des explications profondes sur les mécanismes du C++. Il y a également de nombreux [tutoriels sur le C++](#) qui abordent des aspects spécifiques afin de combler les lacunes de certains cours ou livres sur des points bien précis du langage. Enfin, si vous n'avez toujours pas trouvé de réponse à vos questions, n'hésitez pas à vous rendre sur [le forum C++](#), lieu d'entraide convivial, pour nous faire part de vos difficultés et de vos questions.

Par ailleurs, c'est une mauvaise idée de se lancer dans le C++ en ayant comme objectif immédiat de programmer un clône du dernier jeu à la mode. Bien qu'il y ait de fortes chances qu'il soit effectivement écrit en C++, il a probablement demandé des années de travail à des équipes de programmeurs C++ expérimentés. Il faut apprendre petit à petit, mettre en application ce que l'on apprend, faire un effort de compréhension.

VII - Les bibliothèques

Le C++ n'est pas comme le Visual Basic ou similaire : au début, tout ce que vous aurez, c'est une console. De base, en C++, on ne peut pas ouvrir une fenêtre ou envoyer un email. Il faut pour cela faire appel à ce que l'on appelle des bibliothèques. Ce sont des ensembles de fonctions et classes écrites en C++ qui vous permettront d'effectuer

des tâches précises. Nous vous recommandons ainsi de faire l'effort de vous concentrer sur le langage pendant un moment avant de vouloir créer des applications 3D ou autres clients de messagerie instantanée... Car généralement ces bibliothèques sont écrites par des personnes expérimentées et il n'est pas trivial de s'en servir correctement, malgré les documentations qui les accompagnent. Une fois que vous aurez un niveau satisfaisant en C++, vous pourrez consulter la page [Bibliothèques C++](#) puis la page [Cours sur les bibliothèques C++](#) pour vous former.

VIII - Je voudrais aller plus loin !

Vous aimez le langage et voudriez comprendre plus en profondeur comment tout cela fonctionne ? Vous souhaitez découvrir des techniques avancées, faire en sorte que le langage n'ait plus de secrets pour vous ? Alors voici quelques liens qui vous aideront à acquérir une compréhension profonde du C++.

FAQs

- La [F.A.Q C++](#) aborde également un certain nombre de sujets avancés ;
- [La FAQ Comeau Computing](#), en anglais ;
- [C++ FAQ Lite](#), en anglais, [et sa traduction partielle](#) ;
- [La FAQ de Bjarne Stroustrup, créateur du langage](#) et plus généralement [sa page dédiée aux ressources sur le C++](#), en anglais ;

Articles

- Les [tutoriels C++](#) abordent pour certains des techniques et sujets assez avancés sur le C++ ;
- [Artima > C++ Source](#) : contient de nombreux articles, en anglais, de grands noms du C++ sur des sujets généralement assez avancés ;
- [Dr Dobb's Journal > C++](#) : même chose que le précédent ;

Voici de plus quelques livres qui décortiquent certains aspects du C++ de manière très intéressante et que nous vous recommandons.

Livres

- La série *Effective C++*, *More Effective C++* et *Effective STL* de Scott Meyers : elle aborde des aspects clés du langage C++, tels que la bibliothèque standard, la programmation orientée objet, les exceptions, etc. Le but est de montrer les erreurs à ne pas faire et d'enseigner les bonnes pratiques du C++ ;
- La série *Exceptional C++*, *More Exceptional C++* and *Exceptional C++ Style* de Herb Sutter : sensiblement de même nature que la série d'ouvrages de Scott Meyers ;
- *C++ Coding Standards*, de Herb Sutter et Andrei Alexandrescu : cet ouvrage enseigne les bonnes lignes de conduite en C++, sans trop rentrer dans les détails comme cela est fait dans les deux séries d'ouvrage ci-dessus ;
- *Modern C++ Design* de Andrei Alexandrescu et *C++ Templates : The Complete Guide* de David Vandevorde et Nicolai Josuttis abordent la conception générique en C++ et vous présentent en profondeur les templates du C++ ;
- *Multi-Paradigm Design for C++*, de James O. Coplien : montre comment la combinaison des deux paradigmes principaux de C++, la programmation orientée objet et la programmation générique, s'avèrent puissante et efficace pour le développement C++ quotidien ;

Pour terminer, vous plonger dans du code écrit par d'autres (expérimentés si possible) ne pourra vous être que bénéfique, afin d'élargir votre vision du C++, d'apprendre des petites techniques qui font la différence, etc. Regarder le code de bibliothèques C++ ou bien de petits logiciels ou jeux vous sera ainsi très bénéfique.

IX - Conclusion

Vous avez désormais toutes les cartes en main pour effectuer un apprentissage du C++ sur mesure, efficace. Comme pour l'apprentissage de n'importe quel langage, nous vous conseillons de mettre en pratique autant que nécessaire pour votre compréhension. Enfin, n'oubliez pas que le C++ est riche et par conséquent complexe, donc peut-être ressentirez-vous des difficultés à certains moments ; toutefois, cela fait sa force et c'est pourquoi il est important de surmonter ces difficultés.

Enfin, je tiens à remercier **dourouc05**, **3DArchi**, **koala01** et le reste de l'équipe C++ pour leurs remarques judicieuses et leurs nombreuses relectures.